Computer Networks
ICS 651

- IP routing tables
- IP addressing
- local configuration
- a multiplexed packet network
- distance-vector routing

Sample Routing Table: IPv4

```
% route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         128.171.24.193  0.0.0.0         UG    0      0        0 eth1
10.0.0.0        172.17.70.33    255.0.0.0       UG    0      0        0 eth0
128.171.24.192  0.0.0.0         255.255.255.192 U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1003   0        0 eth1
172.16.0.0      172.17.70.33    255.240.0.0     UG    0      0        0 eth0
172.17.70.32    0.0.0.0         255.255.255.224 U     0      0        0 eth0
192.168.0.0     172.17.70.33    255.255.0.0     UG    0      0        0 eth0
```

- “Gateway” is the IP address of the next hop router
- if there is no gateway (no G flag), the destination address should be on the directly connected network
- the network mask is a dotted-decimal representation of the number of bits in the network part of the address
  - e.g. 255.0.0.0 is an 8-bit network number
  - 255.255.255.192 is a 26-bit network number
  - how many bits does 255.240.0.0 represent?

IPv4 Addresses

two ways of saying which part is the network and which part the host number:

1. class-based: the first few bits tell us how many bits are in the network part (class A: 8 bits, class B: 16 bits, class C: 24 bits). This is the older way of doing this (but is the standard way in IPv6).

2. class-less (newer): each routing table entry also has a **mask**, a 32-bit number of the form `111...1100...00` that has:

  - a 1 bit for every bit of the address that is part of the network number, and
  - a 0 bit for every bit of the address that is part of the host number

- sometimes we use a number (0..30) instead of a 32-bit mask, e.g. 128.171.10.1/255.255.255.0 can be written as 128.171.10.1/24

this is CIDR, Classless Inter-Domain Routing

Classless Interdomain Routing CIDR

- CIDR is a more efficient way of using IP addresses, because you can:
  - have network sizes other than 2^8, 2^{16}, and 2^{24} addresses
  - do multiple hierarchical subdivisions, e.g. 128.171.0.0/16 for routing to UH, and 128.171.10.0/24 for routing within UH
- CIDR was adopted around 1994, due to impending exhaustion of class B addresses
- destination 0.0.0.0 with netmask 0.0.0.0 identifies the default route – every possible address matches this route

Sample Routing Table: IPv6

```
% route -6n
Kernel IPv6 routing table
Destination                      Next Hop                   Flag Met Ref Use If
::/96                            ::                         !n   1024 0      0 lo
0.0.0.0/96                       ::                         !n   1024 0      0 lo
2002:a00::/24                    ::                         !n   1024 0      0 lo
2002:7f00::/24                   ::                         !n   1024 0      0 lo
2002:a9fe::/32                   ::                         !n   1024 0      0 lo
2002:ac10::/28                   ::                         !n   1024 0      0 lo
2002:c0a8::/32                   ::                         !n   1024 0      0 lo
2002:e000::/19                   ::                         !n   1024 0      0 lo
3ffe:ffff::/32                   ::                         !n   1024 0      0 lo
fe80::/64                        ::                         U    256 0      0 eth0
fe80::/64                        ::                         U    256 0      0 eth1
::/0                             ::                         !n   -1  1 45053 lo
::1/128                          ::                         Un   0   3  3794 lo
fe80::250:56ff:feb0:63e/128      ::                         Un   0   1      0 lo
fe80::250:56ff:feb0:173a/128     ::                         Un   0   1      0 lo
ff00::/8                         ::                         U    256 0      0 eth0
ff00::/8                         ::                         U    256 0      0 eth1
::/0                             ::                         !n   -1  1 45053 lo
```

- each address in this table shows the number of bits in the network part of the address:
  - /24 means 24 bits are the network prefix, and 128-24 = 104 bits are the host part of the address

IPv6 Addresses

- RFC 4291, IP Version 6 Addressing Architecture
- for many addresses, 64-bit network prefix and 64-bit interface identifier
- network prefix includes a routing prefix and a subnet ID that add up to 64 bits
- the number of bits in the routing prefix is distributed as part of the routing protocol, as in CIDR

Writing IPv6 Addresses

- 8 groups of 16 bits, each group written as 4 hexadecimal digits
- groups are separated by colons: `:`
- only the significant digits need to be written, e.g.

  `1:2:3:4:5:6:7:8` is a valid IPv6 address
- One sequence of 0 groups can be written as ::

  `::1` is the loopback address

  `fe80::250:56ff:feb0:173a` is a valid address
- `::/0` is the network number of the default route

IP Routing, details

- Frequently, more than one route in the routing table will match a given destination address
  - e.g. the default route matches every address
- if so, the route with the longest network mask is uses
  - this route is called the **longest match** [sic]
- if there are multiple longest matches, the one with the lowest metric is used
- all this applies to both IPv4 and IPv6

Routing Errors

- Routing table has more than one entry for a single destination (this is generally OK)
- A destination might be connected, but not be in the table -- no communication is possible
- A packet is routed in the wrong direction, but eventually gets there (not uncommon, OK)
- A packet is routed in the wrong direction, and either starts to loop or ends up at the wrong place, so the packet is lost -- no communication, packet is discarded when TTL reaches zero

all errors (except physical disconnection) are in the routing table

Local Configuration

- each interface must be given its IP address
- host/router must place all the local routes, next hops, and network masks into the routing table
- host/router must know the address of at least the "default" router
- there may be further configuration for DNS, particularly the IP number(s) of DNS servers

```
% ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.70.36  netmask 255.255.255.224  broadcast 172.17.70.63
        inet6 fe80::250:56ff:feb0:173a  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:b0:17:3a  txqueuelen 1000  (Ethernet)
        RX packets 6404101  bytes 1633307132 (1.5 GiB)
        RX errors 0  dropped 331  overruns 0  frame 0
        TX packets 4336508  bytes 25639721821 (23.8 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
% cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 192.168.10.115
```

Project 1

- each interface has its own IPv6 address
- packets are received on interfaces
  - by the data handler
- check the destination address:
- if it is one of my addresses, or the local broadcast address ff02::1, and the next header identifier is 2, this is a routing packet

  => add the routes to your routing table
- else look up the address in the routing table
- if found, decrement the hop limit and add the packet to the corresponding send queue
- if not found, drop the packet

Summary

- the Internet Protocol is designed to take data end-to-end under a "best effort" model
- IP does not provide:
  - reliability
  - in-order delivery
  - error-free delivery
- the major difference between IPv4 and IPv6 is in the addresses
- routing is easy once the tables are built
- summarizing (routing to networks instead of hosts) helps reduce the size of the tables

A Multiplexed Packet Network

- multiplexing lets us share an expensive resource (a network) among many parties
- packet networks are designed for multiplexing: each sender only "consumes" the network while its packet is in transit
- packet networks are designed for survivability: changing the routing table(s) to use alternate route(s) is all that is required to survive the loss of a link
- the nature of IP supports this survivability: best-effort delivery of connectionless datagrams, reordering is OK, distributed route selection

Routing Tables are Essential

- Each router forwards packets based on the routing table
- Inconsistent routing tables lead to routing loops or packets being discarded
  - even if the hardware works fine, incorrect routing tables mean packets won't be delivered
- Ideally, each route is the shortest path to the destination
  - or one of the shortest paths
- The network is dynamic, so routing tables have to be maintained
  - manual creation of routing tables only works for very small networks

Challenge:
Build Routing Tables Automatically

- automatically construct the routing tables for each router
- each router is configured (manually) with the IP address for each interface
- each router can send a message to the other endpoint of a link, and listen for replies, to find out who it is connected to
- on a broadcast network a router can broadcast or multicast a message, and all other routers on the network will reply
- each neighboring router is connected via a link, which may be shared with other routers (in case of a broadcast network) or may be dedicated (point-to-point links)
- given that each router has information about its own links to neighboring routers, how does this information get to all the other routers in the network?

Distributed Routing Algorithms

- Distance Vector:
  - I know how to reach my neighbors
  - I tell my neighbors they can reach my neighbors through me
  - I tell my neighbors they can reach my neighbors' neighbors through me
  - recurse until everyone is reachable
- Link State:
  - distribute each router's link state to all routers
  - each router independently builds a map of the entire network, and uses it for routing

Distance Vector Algorithm -- Generating Information

- routing table has:
  - destination (perhaps with address mask)
  - interface
  - metric/distance (in hops)
  - next hop (IP address)
- I send my entire table (destinations, masks and distances -- no need to send the next hop) to each neighbor, both periodically, and whenever it changes
- the message that is sent has, for each entry:
  - destination, with address mask
  - metric/distance (in hops)

Distance Vector Algorithm -- Processing Information

- when receiving a routing information message from router R on interface IF, look at each entry (IP/mask, d):
  - set d' = d + 1
  - if IP/mask is not in the table, add (IP/mask, IF, d', R) to the routing table
  - if IP/mask already is in the table with interface IF'' and distance d'', then
    - if either d'' > d' or IF'' = IF, then replace the routing table entry with (IP/mask, IF, d', R)
    - otherwise, ignore this entry
- In-class discussion: what happens if the new IP matches an existing IP but with a different mask?

Distance Vector Example

- My routing table:

Destination	Distance	Port	Gateway (next hop)
A	4	eth0	Q
B	2	tty0	R
X	5	eth1	S

- message from neighbor R on port tty0: (A, 2), (B, 3), (X, 5)
- New routing table:

Destination	Distance	Port	Gateway (next hop)
A	3	tty0	R
B	4	tty0	R
X	5	eth1	S

Issues with Distance Vector

- My routing table:

Destination	Distance	Port	Gateway (next hop)
A	3	tty0	R
B	4	tty0	R
X	5	eth1	S

- suppose the link to router R goes down
- the routes to A and B are unusable and can be deleted
- neighbor S advertises routes to A and B with a cost of 4 and 5, so those are added to the routing table
- unfortunately, neighbor S was simply sending back the routes it heard from this router
- there may be a higher-cost route to A or B, but this will be found

Resolving the issue of routers sending back routes no longer valid

- have a small value of infinity (16 in RIP)
- resend tables whenever they change, to get faster counting-to-infinity
- do not send to neighbor N routes that have N as the next hop (split horizon)
- or, send those routes, but with infinite metric (split horizon with poisoned reverse)
- counting to infinity can still happen if more than two nodes

More Issues with Distance Vector

- There is no way in the algorithm to delete routes
- Solution: routes time out when they are not refreshed within a certain time
- Problem: unless all routers time out simultaneously, that route may still be alive in a neighboring router, which may advertise it back to us (unless we use split horizon)
- this again leads to counting to infinity, but very slowly!